Lecture 7

1. Удамшил гэж юу вэ? (тодорхойлолт, C++/java хэл дээр хэрхэн хэрэгжүүлдэг талаар бичнэ)

Удамшил нь нэг нь үүсгэсэн классын ахин хэрэглэхтэй холбоотой ойлголт ба ингэснээр өртөг, хийх ажлын хэмжээ буурдаг. Эх классаас шинээр класс үүсгэхэд удамшиж байгаа класс эх классынхаа зарим эсвэл бүх шинжийг өвлөн авах ба энийг шинээр класс үүсгэх удамшлыг механизм гэнэ.

C++ Хэлэнд :

Энгийн удамшил, Нийлмэл удамшил, Олон түвшинт, Шаталсан, Хольмог гэж удамшлын 5 хэлбэр байдаг.

Java хэлэнд :

Нийлмэл удамшил хольмог удамшил байдаггүй.

2. Удамшлын горим. public, private, protected горимын талаар тайлбарлаж жишээгээр батална.

Класс удамшуулах хэлбэр нь : `class engineer : public employee`

Класс удамшихад хэзээ ч private горимд байгаа гишүүн шууд private-рүү удамшдаггүй харин 4-дэх горим буюу hidden горим руу удамшина. Энд байгаа мэдээлэл рүү зөвхөн эх классаас удамшиж ирсэн гишүүн функцаар л хандаж болдог.

Private горимруу удамшуулахад protected, public горимд байгаа гишүүнүүд шинэ классын private горимд очиж удамшина.

Protected горимруу удамшуулахад protected, public горимд байгаа гишүүнүүд шинэ классын Protected горимд очиж удамшина.

Public горимруу удамшуулахад , public горимд байгаа гишүүнүүд шинэ классын Public горимд очиж Protected гормид байгаа гишүүнүүд Protected дээр очно.

Өөрөөр хэлвэл горим буурч удамшдаггүй.

3. Удамшлын давуу талуудыг тоочин бичиж бодит жишээн дээр тайлбарла.

Нэгэнт бий болсон зүйлийг ахин хэрэглэх тухай ойлголт учир туршигдаж бэлэн болсон кодыг өөр системд дахин хэрэглэж болох юм. Зүгширсэн классыг өөр програмд аль болох хэрэглэснээрээ хийх ажлын хэмжээ буурч, програмд гарч болох алдааг багасгахаас бөгөөд ингэснээрээ өртөг буурна.

Жишээ нь нэг л шинжийг нь өөрлөн ашиглаж болох бэлээхэн код байхад өөрийнхөө бичсэнийг ахин дахин давтаад байвал утгагай. Software engineer хүн боддог болхоос бичээч биш шүүдээ. DRY гэж үг ч байдаг(Don’t repeat yourself).

4. Удамшлын хэдэн төрөл байдаг вэ? Тус бүрийг тайлбарлан бич.

Удамшлын 5 төрөл C++ хэлэнд байдаг.

> Энгийн удамшил:

Удамших класс зөвхөн нэг эх класстай байна. Өөрөөр бол нэг:нэг буюу дан удамшил.

> Нийлмэл удамшил:

Удамших класс 2-оос цөөнгүй эх класстай бол нийлмэл удамшил болно. Хэд хэдэн эх классын шинжийг хуулбарлан авна.

> Олон түвшинт удамшил:

Удамших класс өөрөө удамших классаас удамшина. $a \rightarrow b \rightarrow c$

> Шаталсан удамшил

Нэг эх классаас хоёроос цөөнгүй удамших класс үүсэх бол ийм удамшлын шаталсан удамшил гэнэ. $a \rightarrow b$ $a \rightarrow c$

> Хольмог удамшил

Энэ нь Нийлмэл удашил ба шаталсан удамшилын нийлбэр юм.

A→B A→C

B:D→C

Бодлого:

```
#include <iostream>
using namespace std;

class shape {
private:
  string name;
  int dimensions;

public:
  shape() {}
  shape(string name) { this->name = name; }
  void setname(string name) { this->name = name; }
  string getName() { return name; }
};

class twoDimensionalShape : public shape {
protected:
  double area;
  double coordinates;

public:
  twoDimensionalShape() {}
  twoDimensionalShape(string name) { setname(name); }
  void setArea(double area) { this->area = area; }
  double getArea() { return area; }
  void setCoordinates(double coordinates) { this->coordinates = coordinates; }
  double getCoordinates() { return coordinates; }
};

class threeDimensionalShape : public shape {
protected:
  double volume;
};

class circle : public twoDimensionalShape {
private:
  double radius;

public:
  circle(string name, double radius, double coordinates) {
    setname(name);
    setCoordinates(coordinates);
    this->radius = radius;
```

```
  }
  void setRadius(double radius) { this->radius = radius; }
  double getRadius() { return radius; }
  double getArea() { return 3.14 * radius * radius; }
  double getPerimeter() { return 2 * 3.14 * radius; }
};
class quadrat : public twoDimensionalShape {
private:
  double side;

public:
  quadrat(string name, double side, double coordinates) {
    setname(name);
    setCoordinates(coordinates);
    this->side = side;
  }
  void setSide(double side) { this->side = side; }
  double getSide() { return side; }
  double getArea() { return side * side; }
  double getPerimeter() { return 4 * side; }
};

class triangle : public twoDimensionalShape {
private:
  double side;

public:
  triangle(string name, double side, double coordinates) {
    setname(name);
    setCoordinates(coordinates);
    this->side = side;
  }
  void setSide(double side) { this->side = side; }
  double getSide() { return side; }
  double getArea() { return (side * side) / 2; }
  double getPerimeter() { return 3 * side; }
};

int main() {
  circle c("circle", 5, 2);
  cout << c.getName() << endl;
  cout << c.getArea() << endl;
  cout << c.getPerimeter() << endl;
  quadrat q("quadrat", 5, 2);
  cout << q.getName() << endl;
  cout << q.getArea() << endl;
  cout << q.getPerimeter() << endl;
  triangle t("triangle", 5, 2);
  cout << t.getName() << endl;
  cout << t.getArea() << endl;
  cout << t.getPerimeter() << endl;

  return 0;
}
```